(In)Efficiency and Reasonable Cost Models

Beniamino Accattoli

INRIA & École Polytechnique

Once Upon a Time

At the beginning of CS different models were proposed:

- Church's λ-calculus;
- Godel's partial recursive functions;
- Turing's machines (TM);
- ...

Church-Turing Thesis: all models are equivalent.

The Models Farm

All models of effective calculability are equivalent but...

Some models are more equivalent than others.

Church himself found TM more effective than λ-calculus.

In which sense TM are more effective than λ-calculus?

Contemporary Perspective:
The cost models of λ-calculus are unclear.

Turing Machines

Turing machines are effective because of self-evident cost models:

- Time: number of machine transitions;
- Space: maximum number of used cells on the tape;

Complexity theory is based on Turing machines.

Reasonable Computational Model

A computational model X is

reasonable

when X and TM can simulate each other

with polynomially bounded overhead in time

(with respect to their time cost models)

Effective Church-Turing Thesis: all models are reasonable.

(alternatively called *extended*, *efficient*, *modern*, or *complexity-theoretic* thesis)

Example: Random Access Machines (RAM) are reasonable.

Effective Thesis and Complexity theory

Consequence of the effective thesis:
(Super)-polynomial classes (*e.g.* P or NP) are model-independent.

Sub-polynomial time is not stable by changing the model.

Founding fathers' skepticism, revisited:

> Is the λ-calculus a reasonable computational model?
>
> Is there a cost model that makes λ-calculus reasonable?

λ-Calculus

Natural cost models for the λ-calculus:

- Time: number of β-steps;
- Space: maximum size of a term during evaluation;

Something is wrong with this naive approach.

This Talk

Explaining the subtleties of time cost models for the λ-calculus.

Focussing on:

the unavoidable nature of the problem.

efficient *vs* reasonable strategies.

Outline

λ-Calculus

Language:

$$t, u, s \quad := \quad x \quad | \quad \lambda x.t \quad | \quad tu$$

β-Reduction:

$$\frac{}{(\lambda x.t)u \rightarrow_\beta t\{x \leftarrow u\}} \text{ (root } \beta) \qquad \frac{t \rightarrow_\beta u}{ts \rightarrow_\beta us} \text{ (@l)}$$

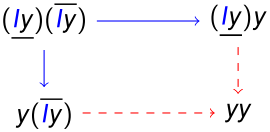
$$\frac{t \rightarrow_\beta u}{\lambda x.t \rightarrow_\beta \lambda x.u} \ (\lambda) \qquad \frac{t \rightarrow_\beta u}{st \rightarrow_\beta su} \text{ (@r)}$$

Confluence

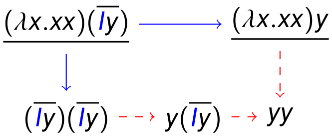
$\rightarrow_\beta$ is non-deterministic but confluent.

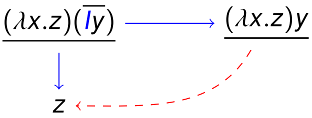
Let $I := \lambda z.z$. Simplest redex $Iy \rightarrow_\beta y$.

Confluence diagrams 1, independent redexes:



Confluence diagrams 2 and 3, duplication and erasure:





Two Main Issues

Two main issues with reasonable time cost models:

- The choice of the evaluation strategy.
- The (non-)atomicity of β-reduction.

A first Look at the Strategy Issue

Confluence = all strategies compute the same result.

Some strategies many terminate while other diverge.

Intuition says that reasonable strategy = efficient strategy.

In particular one expects:

- A reasonable strategy to be terminating;
- A reasonable strategy to take not too many steps.

Both points are misleading!

Atomicity

At first sight, the strategy issue seems more relevant.

But the real issue is the non-atomicity of β-reduction.

Non-atomicity materalizes as the size explosion problem.

Size explosion:
subtle problem, surprisingly neglected by the literature.

Outline

Introducing λ-Calculi

The Structure of the Problem

The Deterministic λ-Calculus

Introducing Size Explosion

Size Explosion is Everywhere

λ-Calculi are Reasonable

Unfolding and Reasonable Representations

Efficiency and Reasonable Cost Models

Higher-Order vs First Order

Two views on computation:

First-Order: programs acting on numbers, strings, etc;

Higher-Order: programs acting on programs.

Turing Machines are first-order.

λ-calculus models higher-order computation.

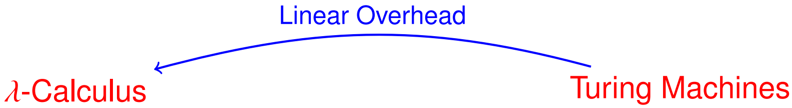
Expected: higher-order reasonably simulates first-order.

Unclear: does first-order reasonably simulate higher-order?

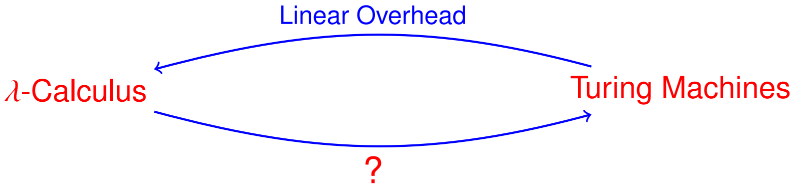
Bird's Eye View

λ-Calculus ? Turing Machines

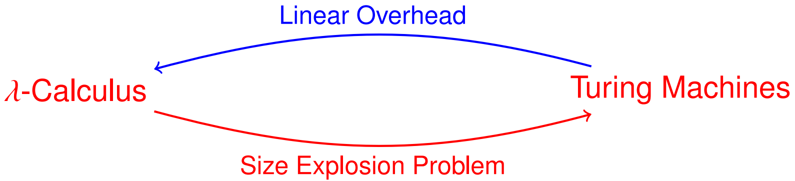
Bird's Eye View



Bird’s Eye View



Bird's Eye View



β-Reduction is Reasonable, Indeed

It turns out that size explosion is circumventable.

The number of β-steps

is

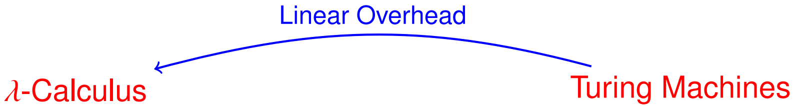
a reasonable time cost model.

First result in a special case in 1995 by Blelloch and Greiner.

General result in 2014 by Accattoli and Dal Lago.

Outline

Bird’s Eye View



From TM to λ-Calculus

First simulation TM $\to$ λ-calculus in 1936 (Turing).

Nowadays, disappeared from the literature

(that rather shows the simulation partial recursive functions $\to$ λ-calculus).

TM $\to$ λ-calculus is the easy direction, and yet subtle.

TM can be represented on a tiny fragment of the λ-calculus.

So tiny, that the strategy problem disappears.

A Simulation Stronger Than It Seems

λ-calculus simulates TMs with linear overhead.

The result is due to Ugo Dal Lago.

It is part of Accattoli & Dal Lago in RTA 2012, in the Appendix.

At the time, we did not pay attention to this very strong fact.

I reworked the simulation, there is a note on my webpage.

The Deterministic λ-Calculus

$\Lambda_{\mathtt{det}}$ is given by the following restricted language:

$$\begin{array}{rcl} t, u, s & ::= & x \mid \lambda x.t \mid t v \\ v & ::= & x \mid \lambda x.t \end{array}$$

endowed with weak evaluation (CbN and CbV coincide):

$$\frac{}{(\lambda x.t)v \rightarrow_{wh} t\{x \leftarrow v\}} \text{(root } \beta\text{)} \qquad \frac{t \rightarrow_{wh} u}{ts \rightarrow_{wh} us} \text{(@l)}$$

In $\Lambda_{\mathtt{det}}$, β-reduction is deterministic.

$\Lambda_{\mathtt{det}}$ is the intersection of the weak and the CPS λ-calculi.

Perpetual Weak Strategies can be Reasonable (!)

Closed $\Lambda_{\mathtt{det}}$ simulates TMs with linear overhead.

In $\Lambda_{\mathtt{det}}$ all weak strategies collapse.

Therefore, every weak strategy simulates TMs efficiently.

Every weak strategy with polynomial overhead is reasonable.

Even if it is perpetual (*i.e.* it diverges as soon as possible)!

A reasonable strategy needs not to be terminating!!!

Outline

Bird's Eye View

Warming Up

Let δ be the duplicator combinator, *i.e.* $\delta := \lambda x.xx$.

Famous divergent term $\Omega := \delta\delta = (\lambda x.xx)\delta \to_\beta \delta\delta$.

So, infinite iterated duplications: $\Omega \to_\beta \Omega \to_\beta \ldots$.

Trivial fact: time complexity $\geq$ space complexity.

Size Explosion — Example

$t_0 := y$ and $t_n := \delta t_{n-1}$, or $t_n := \underbrace{\delta(\delta(\delta \ldots (\delta}_{n \text{ times}}\, y) \ldots))$.

$t_1 = \delta y = (\lambda x.xx)y \to_\beta yy$

$t_2 = \delta t_1 \to_\beta \delta(yy) \to_\beta (yy)(yy)$

$t_3 = \delta t_2 \to_\beta \to_\beta \delta((yy)(yy)) \to_\beta ((yy)(yy))((yy)(yy))$

$\ldots$

$t_n \to_\beta^n y^{2^n}$

Size-Explosion:

The size $|t_n|$ of the initial term is linear in n;

The number of steps $\to_\beta^n$ is linear in n;

The size $|y^{2^n}|$ of the final term is exponential in n.

Size Explosion — The Moral

Time complexity = number of β-steps?

Size-explosion suggests no:

Number of β-steps

does not even account for the time

to write down the result.

Outline

Is It the Strategy?

λ-calculus is non-deterministic but confluent.

Different strategies have very different evaluation lengths.

Does size explosion depend on the evaluation strategy?

No, all strategies suffer from size explosion.

Back to the Atomicity of β

There is an exploding family in the deterministic λ-calculus $\Lambda_{\mathtt{det}}$.

In $\Lambda_{\mathtt{det}}$, all strategies collapse.

Therefore,

No strategy

is immune from

size explosion.

Size Explosion — Worst Case Ever

Consider:

$$s_1 \quad := \quad \lambda x.\lambda y.(yxx)$$

$$s_1 I \quad = \quad (\lambda x.\lambda y.(yxx))I \quad \to_\beta \quad \lambda y.(yII) \quad = \quad r_1$$

Define the following families of terms s_n and exploding results r_n:

$$s_{n+1} \quad := \quad \lambda x.(s_n(\lambda y.(yxx))) \qquad\qquad r_{n+1} \quad := \quad \lambda y.(y r_n r_n)$$

Note that $|s_n| = O(n)$ and $|r_n| = \Omega(2^n)$.

Size Explosion: $s_n I \to_\beta^n r_n$.

Key property:

$$s_{n+1} r_m \quad \to_\beta \quad s_n\,(\lambda y.(y r_m r_m)) \quad = \quad s_n r_{m+1}$$

Outline

λ-Calculus

Natural cost models for the λ-calculus:

Time: number of β-steps;

Space: maximum size of a term during evaluation;

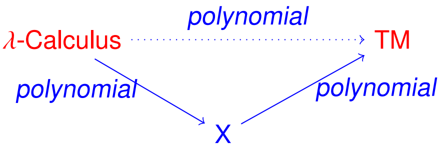
Problem: size explosion.

It looks like the problem is about time.

In fact, it is the other way around... the problem is about space.

Hidden wrong assumption: space = size of the term.

How to Stop Worrying and Love the Bomb



Introduce an intermediate system *X*;

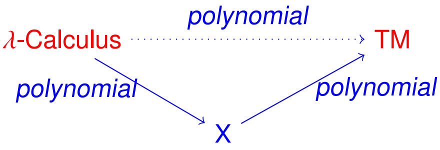
X simulates λ-calculus up to some form of sharing;

X computes a compact representation of the result;

Avoiding size-explosion.

In the best cases the polynomials are linear.

Decomposing the λ-Calculus



In the literature there are 3 instances for *X*:

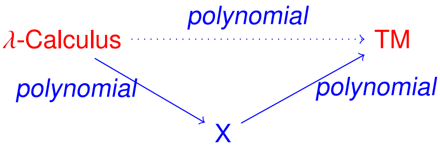
Graph Rewriting (*e.g.* Proof Nets);

Explicit Substitutions (aka `let` expresions);

Abstract Machines.

Fixed *X*, the details also depend much on λ-dialect under study.

Complexity of a Strategy



Fix a strategy $\rightsquigarrow$ and an evaluation $t_0 \rightsquigarrow^n u$.

Implementation in X: $t_0 \rightsquigarrow^n u$ iff $s_{t_0} \rightsquigarrow_X^* s'$ with $\underline{s'} = u$.

Complexity of $\rightsquigarrow$ = cost of implementing $\rightsquigarrow$ in X with $\rightsquigarrow_X$, wrt:

1. Input: size $|t_0|$ of the initial term.
2. Strategy: # of β-steps n.

$\rightsquigarrow$ is a reasonable strategy

=

cost of $s_{t_0} \rightsquigarrow_X^* s'$ is polynomial in $|t_0|$ and n.

Recipe for Proving that a Strategy is Reasonable

$\rightsquigarrow$ is a reasonable strategy

$=$

cost of $s_{t_0} \rightsquigarrow_X^* s'$ is polynomial in $|t_0|$ and n.

3 ingredients:

1. Reasonable micro steps:
   the cost of each $\rightsquigarrow_X$ steps is polynomially bounded.

   $\Rightarrow \rightsquigarrow_X$ is reasonable.

2. Reasonable simulation:
   number of $\rightsquigarrow_X$ steps reasonable in the number of $\beta \rightsquigarrow$-steps.

   $\Rightarrow \rightsquigarrow$ is reasonable.

3. Reasonable representations:
   terms with sharing can be compared without unsharing them.

Outline

Explicit Substitutions & Unfolding

Our implementation scheme for an evaluation $t_0 \rightsquigarrow^n u$:

Input: a λ-term t_0;

Output: a compact representation r of the result u.

Compact representation = a term with Explicit Substitutions.

Real normal form obtained via decoding / unfolding ES.

Unfolding: r is a shared representation of the λ-term $r{\downarrow}$:

$$
\begin{array}{rclcrcl}
x{\downarrow} & := & x; & \quad & (rs){\downarrow} & := & r{\downarrow}\, s{\downarrow}; \\
(\lambda x.r){\downarrow} & := & \lambda x.r{\downarrow}; & \quad & r[x{\leftarrow}s]{\downarrow} & := & r{\downarrow}\{x{\leftarrow}s{\downarrow}\}.
\end{array}
$$

Example:

$$
\begin{array}{rcl}
(xx)[x{\leftarrow}yy][y{\leftarrow}zz]{\downarrow} & = & (xx)\{x{\leftarrow}yy\}\{y{\leftarrow}zz\} \\
& = & (yyyy)\{y{\leftarrow}zz\} \\
& = & zzzzzzzz
\end{array}
$$

ES are a Reasonable Representation

Our implementation scheme for an evaluation $t_0 \rightsquigarrow^n u$:

Input: a λ-term t_0;

Output: a compact representation r of the result $r{\downarrow} = u$.

Size explosion: $r{\downarrow}$ can be exponentially bigger than r.

Are we hiding size explosion in $r{\downarrow}$?

NO! ES are a reasonable compact representation:

Theorem (Accattoli & Dal Lago 2012)
$r{\downarrow} = s{\downarrow}$ can be checked in time polynomial in $|r| + |s|$.

Grabmeyer & Rochel, ICFP '14: $r{\downarrow} = s{\downarrow}$ is pseudo-linear.

Recipe for Proving that a Strategy is Reasonable

3 ingredients:

1. Reasonable micro steps;
2. Reasonable simulation;
3. Reasonable representations (just treated).

Point 3 has to be proved only once.

Points 1 and 2 depend very much on the λ-dialect.

Literature: Closed Cases

In the Closed λ-Calculus (*i.e.* weak evaluation + closed terms):

The number of β-steps is a reasonable cost model.

Ordinary abstract machines (*e.g.* KAM) are enough.

Both reasonable steps and reasonable simulation are easy.

This is enough for the effective Church-Turing thesis.

Literature: Closed Cases

3 independent proofs for the Closed λ-Calculus:

Blelloch & Greiner '96;
(CbV)

Sands & Gustavsson & Moran '02;
(CbV and CbN)

Dal Lago & Martini '09.
(CbV and CbN)

Decomposed and refined by the Accattoli & coauthors in '14.
(CbV, CbN, CbNeed)

Literature: Strong Case

Main result on cost models (Accattoli & Dal Lago, CSL-LICS '14):

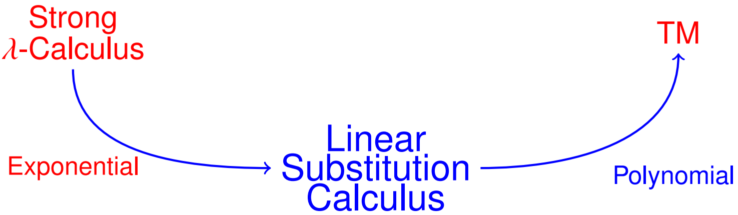
> The leftmost strategy is reasonable.

Reasonable steps is easy.

Reasonable simulation is hard.

Oridnary abstract machines do not work.

Schema of the Solution



Literature: Strong Case

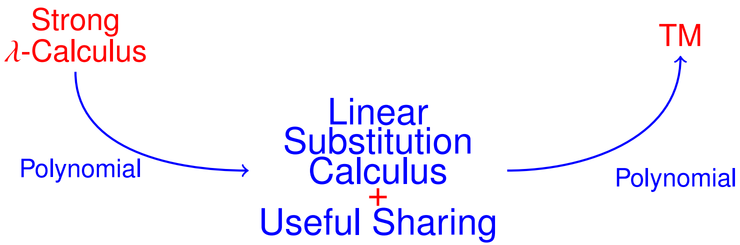
Reasonable simulation requires an additional layer of sharing.

A sophisticated layer called useful sharing.

First proof using LSC, by Accattoli and Dal Lago (2014).

Second proof using an abstract machine, by Accattoli (2016).

Schema of the Solutions



Outline

Literature: Strong Optimal Cases

The sequential optimal strategy is non recursive.

Subtlety: its length can still be a reasonable cost model.

(Is it? Open problem)

Lévy's parallel optimal strategy is recursive and unreasonable.

(Asperti & Mairson 1998)

Intuition:
too many sequential steps merged in a single parallel one.

Open Problem:
is Lévy's strategy efficient? And what does that mean exactly?

Reasonable *vs* Efficient

Efficiency = comparative property.

Reasonable = property of a strategy, in isolation.

Reasonable strategy = implementable with negligible overhead.

Being reasonable is not about efficiency.

Yet, length is an efficiency metric only for reasonable strategies.

Reasonable ⇒ simplified study of efficiency.

Reasonable *vs* Efficient

Leftmost evaluation is desperately inefficient.

Proof that is reasonable requires useful sharing.

Useful sharing is a general, modular technique.

Useful sharing applies to more efficient strategies (CbV / CbNeed).

Reasonable $\Rightarrow$ more efficient implementation techniques.

THANKS!